UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/582,937	06/15/2006	Markus Lauff	2058.102US1	9821

50400 7590 10/28/2010

SCHWEGMAN, LUNDBERG & WOESSNER/SAP
P.O. BOX 2938
MINNEAPOLIS, MN 55402

EXAMINER	
HAYIM, SAMUEL E	

ART UNIT	PAPER NUMBER
2192	

NOTIFICATION DATE	DELIVERY MODE
10/28/2010	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

uspto@slwip.com
request@slwip.com

Office Action Summary	Application No.	Applicant(s)	
	10/582,937	LAUFF ET AL.	
	Examiner	Art Unit	
	SAMUEL HAYIM	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *10 August 2010*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-10 and 12-20* is/are pending in the application.

4a) Of the above claim(s) *3 and 8* is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-2, 4-7, 9-10, 12-20* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

Detailed Action

1. Applicant's amendment and response dated August 10th 2010, responding to the May 26th, 2010, Office Action provided in the rejection of claims 1-10 and 12. Claims 1, 4, 6, 9-10 and 12 have been amended; claims 3 and 8 has been canceled; claim 11 was previously cancelled; and claims 13-20 have been added. Claims 1-2, 4-7, 9-10, and 12-20 are pending in this application and which have been fully considered by the examiner.

Applicant primarily arguing for the claims not being anticipated by *Noble* because *Noble* does not disclose a library having complexity evaluations functions and an aggregator to aggregate the complexity values (*See* paragraphs 1-3 on page 10 of the amendment and response) are not persuasive, as will be addressed under Prior Art's Arguments - Rejections section at item 2 below. Accordingly, the rejection of the claims over the prior art in the previous office action is maintained and **THIS ACTION IS MADE FINAL**. *See* MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply later than SIX MONTHS from the date of this final action.

Prior Art's Arguments – Rejections

2. Applicant's arguments filed on August 10th 2010, in particular pages 9-10, have been fully considered but they are not persuasive. For example:

On pages 9, Applicant contends *Noble* does not disclose a library having complexity evaluations functions. However, Examiner strongly disagrees. (For example, *Noble* page 214-215, a suite of metrics are applied to the GUI in order to determine the overall complexity of a GUI. The suite is collection of complexity functions (as can be further seen by definitions in section 3.1-3.3 where task concordance is computed as the rank order correlation given by Kendall's τ. Layout Uniformity is computed using the function:

$$LU = 100 * \left(1 - \frac{(N_h + N_w + N_l + N_t + N_b + N_r) - M}{6 * N_s - M}\right)$$

. Visual coherence is computed using the function

$$VC = 100 * \left(\frac{\sum_{\forall i} G_i}{\sum_{\forall i} N_i * (N_i - 1)/2}\right)$$

with $G_i = \sum_{\forall j, j \neq i} R_{i,j}$

. The suite of metrics is thereby a library (i.e. a collection of functions).

On page 10, Applicant contends *Noble* does not disclose an aggregator to aggregate the complexity values. However, Examiner strongly disagrees. Examiner notes that Applicant is correct in that is visualized using separate visual encodings, however, for example, Noble page 217 section 4.4. lines 1-2, "the real strength of interactive visualization is that a number of metrics can be displayed at once (Figure 4)" thereby aggregating the metrics into a single display. The fact that each visualized metric has a separate encoding is irrelevant to the

aggregation as the mere display of all visualized metrics at once represents an aggregation of the data).

On page 11, Applicant contends *Noble* does not disclose visually present an aggregated complexity value for each device class the aggregated complexity value comprising a numerical value. The argument is moot in light of new ground of rejection necessitated by amendment.

Claim Objections

1. Claims 13-14 and 17-18 are objected to for lack of antecedent basis. The claims refer to the limitation "child nodes" (claim 13 line 2 and claim 17 line 2) and "parent nodes" (claim 13 and 17 line 2, claim 14 and 18 line 3) which has not been recited in any of the preceding claims.

Claim Rejections - 35 USC § 103

1. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

2. Claims 1-2, 4-7, 9-10, and 12-20 are rejected under 35 U.S.C. 103(a) as being unpatentable over James Noble et al. ("Interactive Design Metric Visualization: Visual Metric Support for User Interface Design" IEEE 1996) (hereinafter Noble) in view of Parker et al. (US

5,600,789) (hereinafter Parker) and in view of Tim Comber and John Maltby ("Investigating Layout Complexity" March 2003) (hereinafter Comber).

As per claim 1, Noble discloses a computer complexity indicator having instructions to evaluate the complexity of a user interface that has device class specific representations, [each device class specific representation referring to a respective device class] and having a respective layout component hierarchy; (For example, abstract, interactive metric visualization is a novel approach providing complex, multi-dimensional feedback on the effects of layout changes in user interface designs; for example, page 213 section 2.1, the layout contains a hierarchy of components that a visualization IDE such as IBM's VisualAge can display. A device class specific representation is a specific user interface. The example of IMB's VisualAge as well as the design implemented by the prior art is intended for use with all user interfaces).

the complexity indicator comprising: a library having complexity evaluation functions to determine complexity values of layout components of the respective layout component hierarchies, where each complexity evaluation fruition is associated with the layout component to which it is applied; (For example, page 213-215 sections 2.2-3.3, a suite (library) of metrics (complexity evaluation functions) are used to determine the complexity values of a corresponding user interface. The complexity evolution functions are associated with each of the components to which the functions are applied (see also figures 1-4)).

and an aggregator to aggregate, using one or more processors, the complexity values by device class according to the corresponding layout component hierarchy of the respective device class specific representation (For example, figure 4, the results are aggregated together to show the relationships between the components (hierarchy) as well as the complexity of the interface

layout. The display pertains to the specific user interface currently under observation but is not limited to just one user interface (i.e. the tool may execute multiple times with each resulting execution associated with a different UI)).

Noble does not expressly disclose each device class specific representation referring to a respective device class.

However, Parker discloses each device class specific representation referring to a respective device class (For example, figure 13 and 15, each operating environment and computer has a separate user interface design that is unique to that respective operating environment or computer).

Noble and Parker are analogous art because they are from the same field of endeavor of testing user interfaces.

It would have been obvious to one of ordinary skill in the art at the time of the invention to modify the user interface testing system as described by Noble and extend it to multiple test multiple user interfaces for different devices as taught by Parker because it would provide for the efficient means for simplifying the ease of testing user interfaces designed for multiple platforms (see Parker column 2 line 26-56).

Noble and Parker do not expressly disclose and a complexity display to visually present an aggregated complexity value for each device class, the aggregated complexity value comprising a numerical value.

However, Comber discloses a complexity display to visually present an aggregated complexity value for each device class, the aggregated complexity value comprising a numerical

value (For example, page 225, table 8 shows the aggregated complexity value for each device class (Scr.1 - Scr.4) as a numerical value).

Noble, Parker and Comber are analogous art because they are from the same field of endeavor of testing user interfaces.

It would have been obvious to one of ordinary skill in the art at the time of the invention to modify the user interface testing system as described by Noble and Parker extend it to define complexity in terms of numerical, graph-able values as taught by Comber because it would provide for the efficient means for determining successful screen design by evaluating complexity (see Comber page 209 introduction).

As per claim 2, Noble discloses the complexity indicator of claim 1, further comprising: a transformer to transform the layout component hierarchy of each representation into a corresponding complexity evaluation hierarchy so that the association of each evaluation function with its respective layout components is redirected through the corresponding component of the respective complexity evaluation hierarchy and the evaluation function is applied to the corresponding component of the respective complexity evaluation hierarchy (For example, figures 2-4, the layout hierarchy is transformed into a complexity evaluation hierarchy (as seen in the figures) where the different lines reflect the metrics (complexity evaluation functions) applied to the different layout components of the complexity evaluation hierarchy).

As per claim 4, Noble discloses the complexity indicator of claim 1, wherein the complexity display [has a drill down portion] to visualize complexity values of layout

components related to a selected device class (For example, figure 4, the figure depicts a complexity evaluation hierarchy based on the selected (currently evaluated) user interface).

Noble does not expressly disclose a drill down portion.

However, Parker discloses a drill down portion (For example, figure 13 and 15, the display of the evaluation of the user interface is based on the selected device (WINDOWS or MAC operating environments (figure 13) or computer 1 or 2 (figure 14)). All devices tested allowing for a selection of one particular devices and then viewing results (i.e. drill down)).

Noble and Parker are analogous art because they are from the same field of endeavor of testing user interfaces.

It would have been obvious to one of ordinary skill in the art at the time of the invention to modify the user interface testing system as described by Noble and extend it to multiple test multiple user interfaces for different devices as taught by Parker because it would provide for the efficient means for simplifying the ease of testing user interfaces designed for multiple platforms (see Parker column 2 line 26-56).

As per claim 5, Noble discloses the complexity indicator of claim 4 in combination with a tree-based outline editor to generate an outline views of the representations that corresponds to the selected device class configured to highlight a layout component that is selected in the complexity display for drill down purposes (For example, figure 3 and page 214 section 3.1, task concordance is a measure of the fit between the expected frequency of various tasks and their relative difficulty using a given interface design. TC is computed as the rank order correlation

between tasks ranked by operational difficulty (e.g. steps or path length) and by anticipated frequency of use. The figure 3 shows an interactive tree-based outline representation of TC which orders the difference tasks based on frequency and pathlength. The tree is based on a selected device class (UI that is currently selected for evaluation).

As per claim 6, Noble discloses a method for complexity evaluation of a user interface, comprising: (For example, abstract, interactive metric visualization is a novel approach providing complex, multi-dimensional feedback on the effects of layout changes in user interface designs).

determining complexity values of layout components of the device class specific representations by applying complexity evaluation functions that are associated with respective layout components; (For example, page 213-215 sections 2.2-3.3, a suite (library) of metrics (complexity evaluation functions) are used to determine the complexity values of a corresponding user interface. The complexity evolution functions are associated with each of the components to which the functions are applied (see also figures 1-4)).

and aggregating, using one or more processors, the complexity values by device class according to a corresponding layout component hierarchy of the respective device class specific representation (For example, figure 4, the results are aggregated together to show the relationships between the components (hierarchy) as well as the complexity of the interface layout. The display p3ertains to the specific user interface currently under observation but is not limited to just one user interface (i.e. the tool may execute multiple times with each resulting execution associated with a different UI)).

Noble does not expressly disclose device class specific representations of the user interface, wherein each device class specific representation referring to a respective device class.

However, Parker discloses receiving device class specific representations of the user interface, wherein each device class specific representation referring to a respective device class; (For example, figure 13 and 15, each operating environment and computer has a separate user interface design that is unique to that respective operating environment or computer).

Noble and Parker are analogous art because they are from the same field of endeavor of testing user interfaces.

It would have been obvious to one of ordinary skill in the art at the time of the invention to modify the user interface testing system as described by Noble and extend it to multiple test multiple user interfaces for different devices as taught by Parker because it would provide for the efficient means for simplifying the ease of testing user interfaces designed for multiple platforms (see Parker column 2 line 26-56).

Noble and Parker do not expressly disclose visually representing an aggregated complexity value for each device class, the aggregated complexity value comprising a numerical value.

However, Comber discloses visually representing an aggregated complexity value for each device class, the aggregated complexity value comprising a numerical value (For example, page 225, table 8 shows the aggregated complexity value, visually represented, for each device class (Scr.1 - Scr.4) as a numerical value).

Noble, Parker and Comber are analogous art because they are from the same field of endeavor of testing user interfaces.

It would have been obvious to one of ordinary skill in the art at the time of the invention to modify the user interface testing system as described by noble and extend it to define complexity in terms of numerical, graph-able values as taught by Comber because it would provide for the efficient means for determining successful screen design by evaluating complexity (see Comber page 209 introduction).

As per claim 7, Noble discloses the method of claim 6, further comprising: transforming the layout component hierarchy of each representation into a corresponding complexity evaluation hierarchy so that the association of each evaluation function with its respective layout component is redirected through the corresponding component of the respective complexity evaluation hierarchy and the evaluation function is applied to the corresponding component of the respective complexity evaluation hierarchy (For example, figures 2-4, the layout hierarchy is transformed into a complexity evaluation hierarchy (as seen in the figures) where the different lines reflect the metrics (complexity evaluation functions) applied to the different layout components of the complexity evaluation hierarchy).

As per claim 8, Noble discloses the method of claim 6, further comprising: visualizing the aggregate complexity values by device class (For example, figure 4, the figure shows the complexity display which is a visualization of the aggregate complexity values).

As per claim 9, Noble discloses the method of claim 8, wherein the visualizing comprises: visualizing complexity values of layout components related to a selected device class

[in a drill down portion] (For example, figure 4, the figure depicts a complexity evaluation hierarchy based on the selected (currently evaluated) user interface).

Noble does not expressly disclose a drill down portion.

However, Parker discloses a drill down portion (For example, figure 13 and 15, the display of the evaluation of the user interface is based on the selected device (WINDOWS or MAC operating environments (figure 13) or computer 1 or 2 (figure 14)). All devices tested allowing for a selection of one particular devices and then viewing results (i.e. drill down)).

Noble and Parker are analogous art because they are from the same field of endeavor of testing user interfaces.

It would have been obvious to one of ordinary skill in the art at the time of the invention to modify the user interface testing system as described by noble and extend it to multiple test multiple user interfaces for different devices as taught by Parker because it would provide for the efficient means for simplifying the ease of testing user interfaces designed for multiple platforms (see Parker column 2 line 26-56).

As per claim 10, Noble discloses a computer system having at least one computing device configured to run an integrated development environment that includes a complexity indicator according to claim 1 (For example, page 213 section 2.1, few design tools also display semantic information about the relationship between the interface and the rest of the program. For example, IBM's VisualAge is one such program which is an integrated development environment (IDE)).

the complexity indicator comprising: a library having complexity evaluation functions to determine complexity values of layout components of the respective layout component hierarchies, where each complexity evaluation function is associated with the layout component to which it is applied (For example, page 213-215 sections 2.2-3.3, a suite (library) of metrics (complexity evaluation functions) are used to determine the complexity values of a corresponding user interface. The complexity evolution functions are associated with each of the components to which the functions are applied (see also figures 1-4)).

an aggregator to aggregate the complexity values [by device class] according to the corresponding layout component hierarchy of the respective [device class specific] representation (For example, figure 4, the results are aggregated together to show the relationships between the components (hierarchy) as well as the complexity of the interface layout. The display pertains to the specific user interface currently under observation but is not limited to just one user interface (i.e. the tool may execute multiple times with each resulting execution associated with a different UI)).

However, Parker discloses device class specific representation (For example, figure 13 and 15, each operating environment and computer has a separate user interface design that is unique to that respective operating environment or computer).

Noble and Parker are analogous art because they are from the same field of endeavor of testing user interfaces.

It would have been obvious to one of ordinary skill in the art at the time of the invention to modify the user interface testing system as described by Noble and extend it to multiple test multiple user interfaces for different devices as taught by Parker because it would provide for the

efficient means for simplifying the ease of testing user interfaces designed for multiple platforms (see Parker column 2 line 26-56).

Noble and Parker do not expressly disclose a complexity display to visually present an aggregated complexity value for each device class, the aggregated complexity value comprising a numerical value.

However, Comber discloses a complexity display to visually present an aggregated complexity value for each device class, the aggregated complexity value comprising a numerical value (For example, page 223, table 4 shows the aggregated complexity value for each GUI as a numerical value).

Noble, Parker and Comber are analogous art because they are from the same field of endeavor of testing user interfaces.

It would have been obvious to one of ordinary skill in the art at the time of the invention to modify the user interface testing system as described by Noble and Parker extend it to define complexity in terms of numerical, graph-able values as taught by Comber because it would provide for the efficient means for determining successful screen design by evaluating complexity (see Comber page 209 introduction).

As per claim 12, Noble discloses a non-transitory machine-readable storage medium storing instruction which when executed by at least one processor provides a method comprising (For example, the software according to the invention executes on a 64M Sparc 4 processor. Processors are coupled to memory (machine-readable medium) that queue up

instructions that enable the processor to execute the instructions in a capable order to produce results).

aggregating, using one or more processors, the complexity values by device class according to a corresponding layout component hierarchy of the respective device class specific representation (For example, figure 4, the results are aggregated together to show the relationships between the components (hierarchy) as well as the complexity of the interface layout. The display pertains to the specific user interface currently under observation but is not limited to just one user interface (i.e. the tool may execute multiple times with each resulting execution associated with a different UI); For example, page 217 section 4.4, the interactive visualization allows for the aggregation of the different metrics to allow for display in a single screen).

determining complexity values of layout components of the device class specific representations by applying complexity evaluation functions that are associated with respective layout components (For example, pages 214-215 section 3-3.4, a suite (library) of metrics (functions) applied to different components of the GUI in determine complexity. Numerical values are obtained from the different metrics).

Noble does not expressly disclose receiving device class specific representations of the user interface, each device class specific representation referring a respective device class

However, Parker discloses receiving device class specific representations of the user interface, each device class specific representation referring a respective device class (For example, figure 13 and 15, each operating environment and computer has a separate user interface design that is unique to that respective operating environment or computer).

Noble and Parker are analogous art because they are from the same field of endeavor of testing user interfaces.

It would have been obvious to one of ordinary skill in the art at the time of the invention to modify the user interface testing system as described by Noble and extend it to multiple test multiple user interfaces for different devices as taught by Parker because it would provide for the efficient means for simplifying the ease of testing user interfaces designed for multiple platforms (see Parker column 2 line 26-56).

Noble and Parker do not expressly disclose visually presenting an aggregated complexity value for each device class, the aggregated complexity value comprising a numerical value.

However, Comber discloses visually presenting an aggregated complexity value for each device class, the aggregated complexity value comprising a numerical value (For example, page 223, table 4 shows the aggregated complexity value for each GUI as a numerical value).

Noble, Parker and Comber are analogous art because they are from the same field of endeavor of testing user interfaces.

It would have been obvious to one of ordinary skill in the art at the time of the invention to modify the user interface testing system as described by Noble and Parker extend it to define complexity in terms of numerical, graph-able values as taught by Comber because it would provide for the efficient means for determining successful screen design by evaluating complexity (see Comber page 209 introduction).

As per claim 13, Noble disclose the complexity indicator of claim 1, wherein the aggregator is to aggregate by propagating the complexity values of child nodes in the layout

component hierarchy to a parent node (For example, figure 3, the two different Print tasks (parent nodes) complexity is shown in figures 3a and 3b. The tasks with the longer lines indicate higher complexity. The longer lines additionally indicate the complexity of the child nodes (subsequent buttons required for the task). The sum total of the child nodes, aggregated, indicate the complexity of the task. Once the buttons are rearranged the task is simpler (figure 3b)).

As per claim 14, Noble discloses the complexity indicator of claim 13, wherein the aggregator is to overrule one or more of the propagating complexity values with a higher complexity value calculated from the parent node (For example, figure 3, the two different Print tasks (parent nodes) complexity is shown in figures 3a and 3b. The tasks with the longer lines indicate higher complexity. The longer lines additionally indicate the complexity of the child nodes (subsequent buttons required for the task). The sum total of the child nodes, aggregated and thereby overruled, indicate the complexity of the task. Once the buttons are rearranged the task is simpler (figure 3b). The sum total is an indication that the task's complexity is a sum of the complexity of the subsequent parts that make up the task and not any one single part).

As per claim 15, Noble discloses the complexity indicator of claim 1, wherein the complexity values comprise numerical values (For example, page 214 section 3, the metric visualization allows for the calculation (numerically) of all complexity values. Task concordance is computed as the rank order correlation, layout uniformity, function is shown in section 3.2 and visual coherence's function is shown in section 3.3).

As per claim 16, Noble does not expressly disclose the complexity indicator of claim 1, wherein the complexity display is to visually present the aggregated complexity value using a graphical bar.

However, Comber discloses wherein the complexity display is to visually present the aggregated complexity value using a graphical bar (For example, figure 3 and figure 8, the graph in figure 8 is a function of application ID and complexity. The graph uses a graphical bar to connect the complexities in order to show relative complexities between GUI's).

Noble and Comber are analogous art because they are from the same field of endeavor of testing user interfaces.

It would have been obvious to one of ordinary skill in the art at the time of the invention to modify the user interface testing system as described by Noble extend it to define complexity in terms of numerical, graph-able values as taught by Comber because it would provide for the efficient means for determining successful screen design by evaluating complexity (see Comber page 209 introduction).

As per claim 17, Noble disclose the method of claim 6, wherein the aggregating comprises propagating the complexity values of child nodes in the layout component hierarchy to a parent node (For example, figure 3, the two different Print tasks (parent nodes) complexity is shown in figures 3a and 3b. The tasks with the longer lines indicate higher complexity. The longer lines additionally indicate the complexity of the child nodes (subsequent buttons required for the task). The sum total of the child nodes, aggregated, indicate the complexity of the task. Once the buttons are rearranged the task is simpler (figure 3b)).

As per claim 18, Noble discloses the method of claim 17, further comprising overruling one or more of the propagating complexity values with a higher complexity value calculated from the parent node (For example, figure 3, the two different Print tasks (parent nodes) complexity is shown in figures 3a and 3b. The tasks with the longer lines indicate higher complexity. The longer lines additionally indicate the complexity of the child nodes (subsequent buttons required for the task). The sum total of the child nodes, aggregated and thereby overruled, indicate the complexity of the task. Once the buttons are rearranged the task is simpler (figure 3b). The sum total is an indication that the task's complexity is a sum of the complexity of the subsequent parts that make up the task and not any one single part).

As per claim 19, Noble discloses the method of claim 6, wherein the complexity values comprise numerical values (For example, page 214 section 3, the metric visualization allows for the calculation (numerically) of all complexity values. Task concordance is computed as the rank order correlation, layout uniformity, function is shown in section 3.2 and visual coherence's function is shown in section 3.3).

As per claim 20, Noble does not expressly disclose the method of claim 6, visually presenting the aggregated complexity value using a graphical bar.

However, Comber discloses wherein the complexity display is to visually present the aggregated complexity value using a graphical bar (For example, figure 3 and figure 8, the graph

in figure 8 is a function of application ID and complexity. The graph uses a graphical bar to connect the complexities in order to show relative complexities between GUI's).

Noble and Comber are analogous art because they are from the same field of endeavor of testing user interfaces.

It would have been obvious to one of ordinary skill in the art at the time of the invention to modify the user interface testing system as described by Noble extend it to define complexity in terms of numerical, graph-able values as taught by Comber because it would provide for the efficient means for determining successful screen design by evaluating complexity (see Comber page 209 introduction).

Conclusion

3. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Samuel Hayim whose telephone number is (571) 270-3370. The examiner can normally be reached on Monday to Friday 8:30 AM to 5:00 PM.

If attempts to reach the above noted Examiner by telephone are unsuccessful, the Examiner's supervisor, Tuan Dam, can be reached at the following telephone number: (571) 272-3695.

The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300. Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status

information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

/SAMUEL HAYIM/
Examiner, Art Unit 2192

/Tuan Q. Dam/
Supervisory Patent Examiner, Art Unit 2192